Lab 2– Playbooks

I- First Playbook :

1. Write your first playbook with a single play running on the group **précédents**.
2. Disable automatic collection of facts.
3. Write a first task that shows you the list of packages to update and test it.
4. Add a task that checks the connection to hosts.
5. Add a fact collection task and a second to display the collected information.
6. Install the **httpd** package. If installation is successful, start the corresponding httpd service.

> touch playbook.yaml
> ansible-playbook playbook.yaml

```
- name: update db servers
  hosts: precedants
  remote_user: root

  tasks:
  - name: ensure postgresql is at the latest version
    yum:
      name: postgresql
      state: latest
  - name: ensure that postgresql is started
    service:
      name: postgresql
      state: started
```

```
---
- hosts: localhost
  become: yes
  gather_facts: no
  tasks:
   - name: check if hosts are accessible
     wait_for:
       host: '{{ (ansible_ssh_host|default(ansible_host))|default(inventory_hostname) }}'
       port: 22
       state: started
       delay: 0
       timeout: 1

   - name: Gather the package facts
     package_facts:
       manager: auto

   - name: Print the package facts
     debug:
       var: ansible_facts.packages

   - name: Print all available facts
     ansible.builtin.debug:
       var: ansible_facts
   - name: install Apache server
     yum:
       name: httpd
```

```
        state: latest
- name: enable and start Apache server
  service:
    name: httpd
    enabled: yes
    state: started
```

II- Control structures:

1. Write a playbook with conditional structures that perform the following tasks (test each task before moving on to the next):

   a. It displays the message "hostname uses the redhat distribution" (resp. debian) if the remote machine's distribution is Redhat or Centos (resp. Debian or Ubuntu); hostname is the remote machine name retrieved from the playbook.

```
- name: Retrieve hostname
  hosts: localhost
  tasks:
    - name: Retrieve the hostname
      command: hostname
      register: result
    - debug:
        var: hostname
```

   b. A second task that displays the message "hostname uses redhat 7 distribution" if the distribution used is Redhat or Centos and version (ansible_distribution_major_version) is 7.

```
- hosts: localhost
  gather_facts: yes
  become: false
  tasks:
  - name: Distribution
    debug: msg="{{ ansible_distribution }}"
  - name: Distribution version
    debug: msg="{{ ansible_distribution_version}}"
  - name: Distribution major version
    debug: msg="{{ ansible_distribution_major_version }}"
```

   c. Add a task that asks the user for their **name** and domain of expertise; if they enter "**devops**" as **expertise**, they will be shown the message "welcome **name**" if not "bye **name**".

```
- hosts: localhost
  vars_prompt:

    - name: username
      prompt: What is your username?
      private: no

    - name: expertise
      prompt: What is your expertise domain?
      private: no
```

```
 tasks:
   - name: Retrieve the hostname
     command: hostname
     register: result
   - debug:
       var: hostname
   - name: Print a message
     ansible.builtin.debug:
       msg: 'Logging in as {{ username }}'
     when: expertise == 'devops'
   - debug:
       msg: 'bye  {{ username }}'
     when: expertise != 'devops'
```

2. Write a playbook with loops **loop**:
    a. Write a task that installs the following packages in a loop :
    - python
    - python-setuptools
    - python-dev
    - build-essential
    - python-pip
    - python-mysqldb

```
- hosts: localhost
  user: ansible
  become: True
  tasks:
    - name: Install all the packages
      apt:
        name: '{{ item }}'
        state: present
        update_cache: True
      with_items:
        - python
        - python-setuptools
        - python-dev
        - build-essential
        - python-pip
        - python-mysqldb
```

```
root@vps290950:~# ansible-playbook installpackages.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [localhost] ***************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************
ok: [localhost]

TASK [Install all the packages] ************************************************************************************************
[DEPRECATION WARNING]: Invoking "apt" only once while using a loop via squash_actions is deprecated. Instead of using a loop to
supply multiple items and specifying `name: "{{ item }}"`, please use `name: ['python', 'python-setuptools', 'python-dev',
'build-essential', 'python-pip', 'python-mysqldb']` and remove the loop. This feature will be removed in version 2.11.
Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
changed: [localhost] => (item=[u'python', u'python-setuptools', u'python-dev', u'build-essential', u'python-pip', u'python-mysqld
b'])

PLAY RECAP *********************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Best solution:

```
- hosts: all
  user: ansible
  become: True
  tasks:
    - name: Install all the packages
      apt:
```

```
    name: "{{ item }}"
    state: present
    update_cache: True
  loop:
    - python
    - python-setuptools
    - python-dev
    - build-essential
    - python-pip
    - python-mysqldb
```

```
root@vps290950:~# ansible-playbook installpackages.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [localhost] ***************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************
ok: [localhost]

TASK [Install all the packages] ************************************************************************************************
ok: [localhost] => (item=python)
ok: [localhost] => (item=python-setuptools)
ok: [localhost] => (item=python-dev)
ok: [localhost] => (item=build-essential)
ok: [localhost] => (item=python-pip)
ok: [localhost] => (item=python-mysqldb)

PLAY RECAP *********************************************************************************************************************
localhost                  : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

>ansible-playbook installpackages.yml -i inventory -l suivants

b. Add 3 groups : orsys, alten, ansible

c. Add 3 users (user1, user2, user3) with secondary groups orsys and alten and with respective passwords ansible, orsys and alten.

```
- hosts: localhost
  user: ansible
  become: True
  tasks:
    - name: Install all the packages
      apt:
        name: "{{ item }}"
        state: present
        update_cache: True
      loop:
        - python
        - python-setuptools
        - python-dev
        - build-essential
        - python-pip
        - python-mysqldb
    - name: Ensure group "orsys" exists
      group:
        name: orsys
        state: present
    - name: Ensure group "alten" exists
      group:
        name: alten
        state: present
    - name: Ensure group "ansible" exists
      group:
        name: ansible
        state: present
    - name: make a new user "user1"
      user:
        name: user1
        state: present
        groups: "orsys, alten"
```

```
      password:
$6$c/CuRmitykVLHst$2lhp0ohRB8m/NDfGDBHWJyVlCGj5bKJD4ECjOMl90KPdvqKlqckkt901beDpElqVkTF8lBeA564j9l9Y1dsQw1
   - name: make a new user "user2"
     user:
      name: user2
      state: present
      groups: "orsys, alten"
      password:
$6$EsFGXVrOT.7d4VJ$uVlT/Aw.5rkRPGc9//Vi3Zvi4M90fA3eyG8mbYfBKxH/5GFNjgmGp810Rnq2Kj9WnsYdYgG9NzOZFj6vwoY5m.
   - name: make a new user "user3"
     user:
      name: user3
      state: present
      groups: "orsys, alten"
      password:
$6$OJSfPQ4aTmWKaW$FxP5o4A2lRYJSDX7WBvIctBBsT5Feua0harI82Qt7v88jWgYSEx2Vhnn36cIZG7bZmYfzcdGeYbz4jEN5BG6w1
```

```
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [localhost] ***************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************
ok: [localhost]

TASK [Install all the packages] ************************************************************************************************
ok: [localhost] => (item=python)
ok: [localhost] => (item=python-setuptools)
ok: [localhost] => (item=python-dev)
ok: [localhost] => (item=build-essential)
ok: [localhost] => (item=python-pip)
ok: [localhost] => (item=python-mysqldb)

TASK [Ensure group "orsys" exists] *********************************************************************************************
ok: [localhost]

TASK [Ensure group "alten" exists] *********************************************************************************************
ok: [localhost]

TASK [Ensure group "ansible" exists] *******************************************************************************************
ok: [localhost]

TASK [make a new user "user1"] *************************************************************************************************
changed: [localhost]

TASK [make a new user "user2"] *************************************************************************************************
changed: [localhost]

TASK [make a new user "user3"] *************************************************************************************************
changed: [localhost]

PLAY RECAP *********************************************************************************************************************
localhost                  : ok=8    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

3. We will now use the menu **vars_files**:

   a. Créer un fichier appelé **packages** containing the list of packages.

```
>touch packages.yml
packages_to_install:
 - python
 - python-setuptools
 - python-dev
 - build-essential
 - python-pip
 - python-mysqldb
```

   b. Create a file called **groups** containing the list of groups g1, g2 and g3.

```
[usera@lx-14-100 ~]$ cat groups.yml
groupes:
 - g1
 - g2
 - g3
```

   c. Create a file called **users** containing the list of users orsys, alten and (your name).

```
>touch  users.yml
---
user_passwords:
```

orsys: 6rounds=656000$m/qpgaPV9nDhZA84$0Uz2fQ7PjnX.eMIDSlw0hUetHYat.VuxIzBNsbceZjg60XMe.0hrDekRybNAMe0fPqvczikY0Hdph8KMhcHct. # ws#P)Bg)l853

alten: 6rounds=656000$RhhaEkZK/60KAYDf$U/nsycrW2A4SAuhBbAW4na4OLunPrUfR31OU3ThY1ge3vc.RUfhyHTg5dShkTYFGB/455lv0vOWDAmbGiOI730 # qbbw8&OeZ1ql

ahmed: 6rounds=656000$aXLv86ermeammjFO$MooGjguTxUjhc2m6OefDddz0mszG/SprKiyTsND0lpT3f4.R7V5KucdK9JdLluOF.WnpGAz/GKy2umf5TPkPr. # zlPjxwCFm@ES

d. Rewrite a second version of the playbook using the tag **vars_files**: to install packages, add users and groups.

> ansible-playbook var_packages.yml --extra-vars "ansible_sudo_pass=usera"

```
- hosts: localhost
  user: ansible
  become: True
  tasks:
    - name: Include variables
      include_vars: 'packages.yml'

    - name: "Get installed packages"
      yum:
        list: "installed"
      register: installed_packages

    - name: "Install missing packages"
      package:
        state: "present"
        name: "{{ item }}"
      with_items: "{{ packages_to_install | difference(installed_packages | json_query('results[*].name')) }}"
```

> ansible-playbook var_users.yml --extra-vars "ansible_sudo_pass=usera"

```
- hosts: localhost
  user: ansible
  become: True
  tasks:
    - name: Load passwords from vault
      include_vars: users.yml

    - name: Create users
      user: name="{{item.key}}" password="{{item.value}}" shell="/bin/bash" update_password=on_create
      with_dict: "{{ user_passwords }}"
      register: user_results

    - name: Force password renewal for newly created users
        command: chage -d 0 {{item.item.key}} # item.item is the key/value pair from the dict in the previous task
      when: "{{item.changed == true}}"
      with_items: "{{ user_results.results }}
```

- ansible-playbook var_groups.yml --extra-vars "ansible_sudo_pass=usera"

```
---
- hosts: localhost
  user: ansible
  become: True
  tasks:
  - name: Load groups from file
    include_vars: "groups.yml"
  - name: Ensure groups exists
    group:
      name: "{{ item }}"
      state: present
```

```
  with_items: "{{ groupes }}"
```

e. Separate the package installation and group creation tasks on two separate files in a tasks folder and rewrite the playbook with the include tag to include the exported tasks.

>ansible-playbook grouped_tasks.yml --extra-vars "ansible_sudo_pass=usera"

```
cat grouped_tasks.yml
---
- hosts: localhost
  user: ansible
  become: True

- include: var_groups.yml
- include: var_packages.yml
```

4. Tag each task in the playbook to: packages, groups and users. Then run only the first 2 tags.

>ansible-playbook playbook.yml –tags "packages, groupes" --extra-vars "ansible_sudo_pass=usera"

III- Role Management:

Roles are an additional level of abstraction in Ansible. Defining roles allows for subsequent combination and “standardization” of a playbook presentation. We will turn our playbook into a role and then quickly see Galaxy, the hub of Ansible where users publish their roles.

1. Write a role

Let's tidy up a bit. We're going to add a "roles" directory which will store the roles. We then create a "gestion" subdirectory which will contain our "gestion" role and different subdirectories:

```
mkdir roles
cd roles
mkdir gestion
cd gestion
mkdir files handlers meta templates tasks vars
```

Next, we will go into tasks where we will put the tasks section into a main.yaml file.

First simplification: in tasks, there should be only tasks and no more headers. In addition, with roles, there is no need to give directory names for files and templates because they are implicit. The variables will also be put in a file under the vars folder.
You then need to copy the index.html file into the correct directory.

We then go over the roles directory and create a file play.yml

This file will use the role management that we just created as follows:

```
- hosts: suivants
  become: yes
  roles:
- role: gestion
```

Then execute play.yml using the command

```
ansible-playbook play.yml
```